2024 18th International Conference on Automatic Face and Gesture Recognition (FG)

Boosting Gesture Recognition with an Automatic Gesture Annotation Framework

Junxiao Shen[1,2], Xuhai Xu[1,3], Ran Tan[1], Amy Karlson[1], and Evan Strasnick[1]
[3] Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology
[2] Department of Engineering, University of Cambridge
[1] Reality Labs Research, Meta

***Abstract*— Training a real-time gesture recognition model heavily relies on annotated data. However, manual data annotation is costly and demands substantial human effort. In order to address this challenge, we propose a framework that can automatically annotate gesture classes and identify their temporal ranges. Our framework consists of two key components: (1) a novel annotation model that leverages the Connectionist Temporal Classification (CTC) loss, and (2) a semi-supervised learning pipeline that enables the model to improve its performance by training on its own predictions, known as pseudo labels. These high-quality pseudo labels can also be used to enhance the accuracy of other downstream gesture recognition models. To evaluate our framework, we conducted experiments using two publicly available gesture datasets. Our ablation study demonstrates that our annotation model design surpasses the baseline in terms of both gesture classification accuracy (3-4% improvement) and localization accuracy (71-75% improvement). Additionally, we illustrate that the pseudo-labeled dataset produced from the proposed framework significantly boosts the accuracy of a pre-trained downstream gesture recognition model by 11-18%. We believe that this annotation framework has immense potential to improve the training of downstream gesture recognition models using unlabeled datasets.**

I. INTRODUCTION

Hand gesture recognition is a widely applied technology in various fields such as augmented reality (AR) [35], and virtual reality (VR) [39], [43]. Various modalities of hand gesture data can be utilized for recognition, such as RGB, optical flow, depth, IR, IR-disparity, and 2D/3D skeletons. Contemporary AR and VR head-mounted devices (HMDs) track the user's hands, providing access to real-time 3D hand skeleton data. In this paper, our focus is on using 3D hand skeleton data. This choice is motivated by the widespread availability of 3D hand skeleton data in modern HMDs and its compatibility across different sensors and devices [43].

A typical pipeline to build a robust gesture recognition model usually starts with data collection, followed by data annotation and model training. The annotation step involves two steps: (1) marking the class of the gesture, and (2) localizing the time range of a gesture. State-of-the-art gesture recognition methods are primarily based on deep-learning techniques [23], [16], which require a significant amount of labeled data for training [31]. Insufficient training data can result in overfitting or a failure to learn for a deep neural-network model [42]. Unfortunately, annotating data can be both time-consuming and costly, especially for open-world gesture recognition [46]. Additionally, many state-of-the-art gesture recognition methods require strongly-segmented data (frame-wise segmentation, i.e., Step 2 of annotation) for high-quality training [16], [4], [23], which needs more intense human manual efforts and adds another layer of expense to the data annotation process.

Previous research has focused on addressing the issue of data sparsity in gesture recognition. Various techniques for data augmentation have been proposed, such as the use of a Generative Adversarial Network (GAN) for realistic transformations on gesture skeleton data [42], [44], and the use of unrealistic distortions like cutout [6] and mixup [61] to regularize neural network training. Another recent work by Xu et al. [56] leveraged a few-shot learning framework to reduce the dataset size requirement. However, while these techniques help alleviate the issue of data sparsity, they all rely on an existing annotated dataset, which still suffers from the high cost of data annotation. There is very little research on *automating the gesture annotation process* that can achieve the two annotation steps simultaneously [19], [14]. Meanwhile, the efficient use of rich unlabeled datasets is underexplored.

To address the gaps, we develop a novel automatic gesture annotation framework (see the red borders in Figure 1). Our framework contains two components: 1) an annotation model (see Figure 2) that enables the prediction of unlabeled gesture data in both gesture classification (step 1 of annotation) and gesture nucleus localization (step 2 of annotation)[1]. 2) a semi-supervised learning pipeline that uses the annotation model (i.e., the first component) to predict unlabeled data and then uses these predictions as pseudo-labels to further augment the annotation model itself. These high-quality labels can then be used to further fine-tune any downstream gesture recognition models (see the upper part of Figure 1).

By combining these components, we have developed a robust and automated framework for annotating gesture data of high quality. This framework enables large-scale annotation without the need for extensive human labor. It is important to emphasize that our framework primarily focuses on the annotation step, which serves as a crucial support for training downstream gesture recognition models in real-time systems.

[1]A gesture is composed of three phases: preparation phase, nucleus, and retraction phase. Gesture localization is defined by determining the temporal position of the gesture nucleus, which is the core part of gesture recognition.

979-8-3503-9494-8/24/$31.00

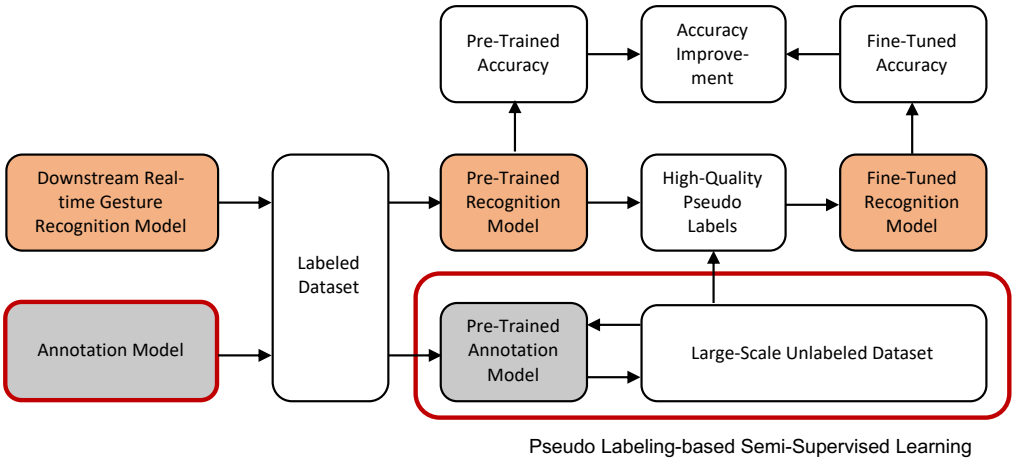

Fig. 1: Our framework provides an automatic gesture annotation solution. The red borders highlight our main contribution. The framework consists of two components: (1) a novel annotation model that utilizes Connectionist Temporal Classification (CTC) loss (see Figure 2), and (2) a semi-supervised pipeline that improves the model's performance by training on its own predictions, i.e., pseudo labels. In real-life (open-world), the gesture annotation framework operates as follows: first, a real-time gesture recognition model and our proposed annotation model are pre-trained on a small labeled dataset. Next, the annotation model is integrated into the pseudo-labeling process, where it produces pseudo labels by annotating an unlabeled dataset, which is then used to augment the annotation model. After pseudo-labeling is complete, the final high-quality pseudo-labels are used to fine-tune the pre-trained real-time gesture recognition model.

To verify the effectiveness of our framework, we carried out an in-depth analysis using two public gesture-related skeleton-based datasets: SHREC'2021 [3] and Online DHG [5]. Initially, we conducted an ablation study to assess the contribution of our different design blocks within the annotation model on gesture classification and nucleus localization tasks. The results showed our model outperforms the baseline model by an improved gesture classification accuracy of 4.3% (3.4%) and an improved gesture nucleus localization accuracy of 71.4% (75.0%) in SHREC'2021 dataset (Online DHG dataset). Next, we used our annotation framework to label a subset of the public datasets mentioned above. We then used this pseudo-labeled dataset, created by our framework, to fine-tune a pre-trained downstream gesture recognition model. The evaluation showed that this pseudo-labeled dataset greatly enhanced the accuracy of the pre-trained downstream gesture recognition model by 11-18%.

To the best of the author's knowledge, this is the first instance of proposing an automatic and semi-supervised gesture annotation framework that performs gesture classification and nucleus localization simultaneously.

II. Related Work

We first briefly overview the related work of gesture recognition and localization. We then summarize the existing work on gesture annotation. We also introduce semi-supervised learning background.

A. Gesture Recognition with Deep Learning

Recently, deep learning techniques have become increasingly popular for gesture recognition. Recurrent neural networks (RNNs) have demonstrated effectiveness in identifying spatial and temporal relationships in activity recognition, both with RGB-based and skeleton-based input [24], [45]. However, when there is a large gap between the input data and the target task, conventional RNNs are not capable of extracting relevant information. To address this issue, LSTMs have been proposed to manage 'long-term dependencies'[12], [27], [41]. Additionally, graph and manifold learning has also achieved success in gesture recognition, as seen in DG-STA [4] and ST-TS-HGR-NET [32]. There has also been a growing interest in gesture recognition in AR and VR applications, such as 3D hand pose estimation using an infrared camera [33] and fast recognition of foot gestures for virtual locomotion [47].

Different modalities of hand gesture data can be used as input for gesture recognition, such as RGB, optical flow, depth, IR-left, IR-disparity, time-series data (IMU, EMG, etc.), non-optical 2D data (tomography, acoustic spectrograms, etc.), and 2D/3D skeletons [52], [59], [13], [55], [29], [26], [21], [51], [9]. The framework proposed in this paper can be easily adapted to different input types. As an illustrating example, this paper uses skeleton data, which records the 2D or 3D positions of key points/joints on the hand. With the advancement of hardware (e.g., Microsoft HoloLens, Intel RealSense, and Leap Motion Controller), the use of skeleton data is becoming increasingly popular among different platforms and modalities. These devices provide precise skeletal data of the hand and fingers in the form of a full 3D skeleton.

B. Gesture Localization

Previous studies have explored gesture localization using RGB data, with early works such as [8], [28], [37] primarily

focusing on spatial segmentation, which involves separating the hand from its surroundings. However, it is important to note that temporal segmentation, specifically determining the timing of the gesture nucleus, is equally important in addition to spatial segmentation. This temporal aspect of gesture localization is commonly referred to as "nucleus localization." Existing approaches often employ heuristic methods to locate gestures, such as the use of sliding windows to analyze the sequence of output confidence/loss from a gesture recognition model and identify peaks or valleys [57], [58]. However, our experiments have shown that such heuristic methods face challenges when dealing with variations in gestures, as depicted in Figure 3.

C. Gesture Annotation

Traditionally, manual annotation software has been employed to annotate gestures, with human annotators manually detecting the start and end points of each gesture. However, this process is time-consuming and requires significant labor. As a result, there has been growing interest in developing automatic annotation tools. While various fields have explored this idea (e.g., [60], [34], [7]), relatively fewer works have focused on gesture data specifically. One relevant work by Kratz et al. [19] presents preliminary research on automatically segmenting motion gestures tracked by IMUs. They suggest that by recognizing gesture execution phases from motion data, it might be possible to automatically identify the start and end points of gestures. However, their work primarily focuses on gesture localization and does not address gesture classification. More closely related to our work, Lenaga et al.[14] propose the use of an Active Learning (AL) [36] framework to automatically detect sign language gesture occurrences in RGB videos. Their approach requires manual annotation for only a small subset of the videos, benefiting researchers studying multimodal communication. However, their primary application area is sign language gestures in RGB videos, which predominantly focuses on spatial data. In contrast, our framework focuses on temporal data for gesture annotation. Furthermore, our approach leverages a semi-supervised learning pipeline to enhance the performance of annotation, setting it apart from previous works.

D. Semi-Supervised Learning Methods

In recent years, there has been a growing interest in semi-supervised Learning (SSL) due to its ability to leverage large amounts of unlabeled data. SSL becomes particularly valuable when labeled data is scarce or when the labeling process is time-consuming and labor-intensive. Consistency regularization [40], [1], [20] and pseudo-labeling [22], [54], [38] are two popular methods utilized to make effective use of unlabeled data, and they have been integrated into various modern SSL algorithms [49], [2], [53].

The semi-supervised learning pipeline within our framework draws inspiration from the recently developed FixMatch algorithm [48]. FixMatch combines these techniques with both weak and strong data augmentations, yielding remarkable results in SSL tasks.

III. Problem Formulation

This section formulates gesture annotation and gesture recognition and discusses their difference. We also explain why simply using a conventional gesture recognition model does not suit data annotation and why an annotation model is necessary.

It is important to clarify that when referring to gesture recognition, we specifically focus on *real-time* gesture recognition. In this context, *real-time* gesture recognition involves identifying gestures within a realistic sequence that includes multiple gesture classes as well as background activities (non-gestures), commonly observed in a sensor stream.

To achieve *real-time* gesture recognition, a typical approach is to utilize a sliding-window-based technique. At each sliding step, a window is fed into the model, producing class-conditional probabilities. The predicted gesture class within the window is determined by taking the maximum value from the probabilities. However, the conventional real-time gesture recognition model is not well-suited for data annotation, particularly for gesture nucleus localization. Post-processing techniques need to be employed to localize the gesture nucleus. However, these post-processing steps often rely on heuristic methods (e.g., [43]), which may be susceptible to variations in gestures and contextual differences.

Furthermore, although both gesture annotation and real-time gesture recognition aim for accurate gesture classification, they have different objectives. While real-time gesture recognition models prioritize detecting gestures as quickly as possible, gesture annotation additionally focuses on identifying the nucleus of the gesture as accurately as possible.

In contrast, an annotation model can incorporate gesture nucleus localization as an inherent training objective since it does not need to optimize for recognition latency. This enables better architectural design choices to improve the performance of the annotation task.

IV. Proposed Framework

In this section, we describe our framework for annotating gesture data.

A. Design Questions

We propose an annotation model with individual components that are specifically designed to optimize two goals of gesture annotation: gesture classification and gesture localization. We start by answering four important model design questions of our architecture.

1) **Small Window Size *vs* Large Window Size:** The window size for a classification model is typically kept small to ensure accurate classification, as it is believed that a window should only be large enough to contain one gesture [43] such that the input data is clearly segmented and the model can be better trained. Another main reason for using a small window size is to achieve low recognition latency [30], which is not a primary

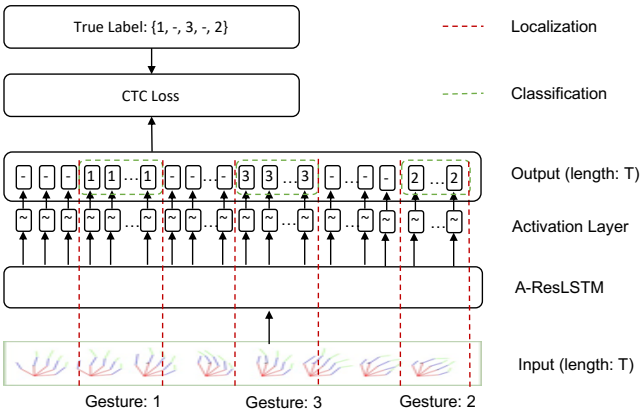

Fig. 2: Due to the extended input window of the annotation model, the true label has the capacity to encompass multiple gestures simultaneously. In the illustrated case, the true labels are {1, - , 3, - ,2}, where "-" represents background activities (*no gesture*). The output of the annotation model is a sequence of predicted labels that has the same length as inputs. The loss function for the proposed annotation model is a Connectionist Temporal Classification (CTC) loss. The backbone of the model is an A-ResLSTM we adopted from Shen et al. [43].

concern during the data annotation process. Moreover, compared to a larger window, a small window contains less information about previous timestamps and has a narrower vision for gesture localization. Therefore, we are using a sliding-window technique with a larger window size to process multiple gestures in the input sequence.

2) **One Prediction Per Frame *vs* Multiple Predictions Per Frame:** When it comes to identifying gestures in a sliding-window approach, a classification model typically outputs a single vector of class-conditional probabilities for all gesture classes (plus *no gesture*). If the slide step is as small as one frame, the classification model can output a single vector per frame. This is a typical many-to-one mapping, as the input is a sequence of vectors, and the output is one vector. However, such a many-to-one mapping loses the fine-grained temporal information by collapsing the multiple frames into one single output. This will impact the gesture localization task. In contrast, a many-to-many (many2many) architecture of each time window contains richer temporal information and allows more dynamic and flexible adjustment between multiple windows. Therefore, we use a many-to-many model to instantiate such architecture. The model output is a sequence of class-conditional probabilities and can be decoded as a sequence of predicted classes, with overlapping predictions from consecutive windows. This allows for dynamic adjustments of earlier predictions based on later windows, and helps to reduce noise in the final prediction.
3) **Strongly-Segmented *vs* Weakly-Segmented Training Data:** It is important to note that a classification model that is trained using cross-entropy loss typically requires clearly defined and segmented gesture data. This means that the start and end frames of the gesture must be accurately labeled. As a result, the model requires strongly-segmented training data. However, strongly-segmented data requires extensive human labor. We thus desire a model that can be trained using weakly-segmented data, while still being able to learn the temporal and spatial properties of the gesture nucleus among the data. Therefore, to handle the challenge of the need for strongly-segmented data, we adopt a Connectionist Temporal Classification (CTC) loss, which can automatically align the unsegmented input sequence with the output sequence. This removes the requirement for strong segmentation of gestural sequences. This loss has been widely adopted in seq2seq model training [10].
4) **Heuristic Parameters *vs* Automation:** Shen et al. [43] used heuristic threshold parameters to estimate gesture locations. However, the accuracy of such an estimation largely depends on the other parameters of the model (window length, slide step etc.). This is because the training label does not directly contain any positional information about the gesture nucleus, and such a conventional real-time gesture recognition model is not explicitly designed for this type of localization. Therefore, we desire a model that does not need these heuristic parameters. This is another reason supporting our choice of the CTC loss. The spiky nature of the CTC's output can eliminate the need for post-processing [50], [18], and allow for the localization of the gesture nucleus without the need for any heuristic parameters. We can identify the nucleus position directly by identifying the spike position. Moreover, greedy search can be used to decode the model output into the sequence of gesture classes.

B. Annotation Model

After answering these four design questions, we now introduce our model's individual components in detail.

1) Seq2seq Model: We design a seq2seq model that is a many2many model. It has an input window $\mathbf{W}$ of frame length L (window size) and an output which is a matrix of probabilities $\mathbf{M} \in \mathbb{R}^{L\times(K+1)}$. It defines the probabilities of detecting a gesture (or *no gesture*) k at time t in an input window $\mathbf{W}$: $p(k,t \mid \mathbf{W}) = \mathbf{M}_k^t \forall t \in [0,N)$. The backbone of our seq2seq model is an A-ResLSTM model we adopted from Shen et al. [43], which is one of the state-of-the-art neural network architectures for gesture recognition. This network is composed of residual blocks [11] and bi-directional LSTM layers [12] with an attention layer [25].

2) Connectionist temporal classification (CTC): Connectionist temporal classification (CTC) is a type of neural network output that can be constructed into a loss function [10]. It is designed to tackle the alignment problems between two sequences with very different lengths, such as in handwriting recognition where the written text

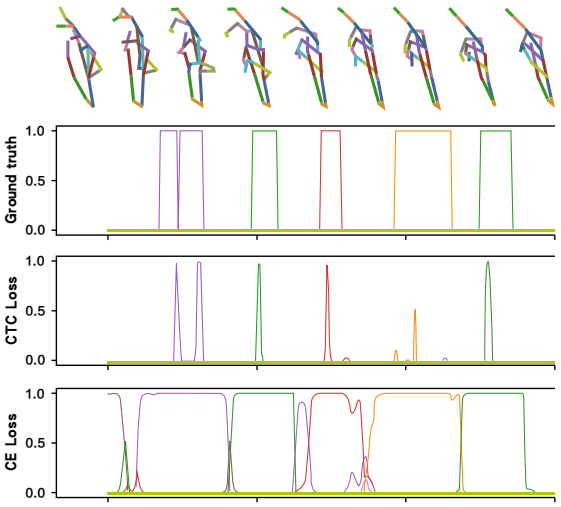

Fig. 3: Demonstration of the network output probabilities from a CTC and CE (cross-entropy) trained network versus the ground truth [50]. The spike of CTC loss clearly captures the gesture nucleus, while the curve of CE loss needs post-processing. For example, the CTC loss successfully distinguishes two consecutive same gestures, while the CE loss confuses them together.

is often longer than the number of characters. This is achieved by introducing a pseudo-character (called blank $-$, or *no label*) to encode duplicate characters in handwriting recognition. In the gesture recognition case, the pseudo-character is the same as *no gesture*. We can then condense repeated class labels and remove *no gesture* labels, e.g., $[-,-,-,1,1,1,-,2,-,3,3,3,-,-] = [1,1,1,-,-,-,2,2,-,-,3,-] = [1,2,3]$, where 1, 2, 3 are actual gesture classes and "$-$" is *no gesture*.

Mathematically, we define a path π as a possible mapping of the input sequence $\mathbf{W}$ of length L into a sequence of training class labels $\mathbf{y}$. The probability of observing path π is $p(\pi \mid \mathbf{W}) = \prod_t \mathbf{M}^t_{\pi_t}, \forall \pi \in A'^L$, where π_t is the class label predicted at time t in path π, and A'^L is the set of length L sequences (paths π) over the gesture dictionary $A' = A \cup \{\textit{no gesture}\}$. The next step is to define a many-to-one mapping operator $y = \gamma(\pi)$ to map the paths into a sequence of gesture labels, $\gamma : A'^L \mapsto A^{\leq L}$, after condensing class labels and removing *no gesture* labels as previously noted. Thus many paths π under γ result in the same gesture sequence $\mathbf{y}$. The probability of observing the gesture class sequence $\mathbf{y}$ given an input window $\mathbf{W}$ is the sum of the condition probabilities of all path π mapping to that sequence, $\gamma^{-1}(\mathbf{y}) = \{\pi : \gamma(\pi) = \mathbf{y}\}$:

$$p_{CTC}(\mathbf{y} \mid \mathbf{W}) = \sum_{\pi \in \gamma^{-1}(\mathbf{y})} (p(\pi \mid \mathbf{W})) \tag{1}$$

Therefore, the CTC loss is:

$$L_{CTC} = -log(p_{CTC}(\mathbf{y} \mid \mathbf{W})) \tag{2}$$

expressed in the log domain. $p_{CTC}(\mathbf{y} \mid \mathbf{W})$ can be calculated efficiently using a dynamic programming algorithm [10]. Derivatives of the loss can then be calculated for backpropagation.

Compared to conventional cross-entropy (CE) loss, CTC loss has the benefit of producing output with distinct, sharp predictions that can accurately capture consecutive gestures. In contrast, a model trained with CE tends to blend predictions of the same class together (see Figure 3). This means that in order to correctly identify when one gesture has been performed, the output from a CE-trained classification model must be processed further with some heuristic methods with errors (e.g., a single-time activation algorithm [43]). However, with CTC, the sharp nature of the predictions automatically eliminates such error-prone step [50], [30], [10].

3) Dynamic Adjustment: Due to the sliding-window fashion, for a window $\mathbf{W}$ of length L sliding at step N, the model outputs $\mathbf{M}$ for each window. $\mathbf{M}$ consists of L vectors of class-conditional probabilities $\mathbf{p}$. Therefore, the model will produce L/N times class-conditional probabilities $\mathbf{p}$ at each step/frame, forming a sequence $(\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_{L/N})$. We perform a dynamic adjustment on labeling and nucleus localization. Specifically, we average these predictions to form one vector of class-conditional probabilities for each step, i.e. $\mathbf{p}_{average} = \text{average}(\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_{L/N})$ We determine the location of the gesture's nucleus by identifying the position of the highest point in the gesture's prediction.

4) Decoding: There are different decoding algorithms for the dynamic adjustment output. One option is greedy search, which approximates the solution by taking the most likely class at each step in the matrix $\mathbf{M}$. It is efficient to compute as it simply concatenates the most active outputs at each step. Another option is beam search. It uses more information from the output sequence by expanding all possible next steps and keeping track of the k most likely steps, where k is the beam factor, or the maximum number of complexes to be specialized. It requires more memory and computational power as k increases. But it can generate multiple sequence candidates, which allows for selection based on other contextual information, such as a user's posture. However, as there is currently no gesture dataset containing such context information, we leave the exploration of beam search or other decoding methods as future work and use greedy search in this paper.

Our newly proposed annotation model enables accurate gesture annotations on unlabeled datasets, including both classification and localization. These annotations can be utilized to train various downstream gesture recognition models. However, we also recognize that the potential of unlabeled datasets can be further leveraged. To address this, we introduce a semi-supervised learning pipeline that enhances the performance of both the annotation model and downstream gesture recognition models. This pipeline maximizes the benefits of unlabeled data and improves the overall performance of the framework.

C. Semi-Supervised Learning Pipeline with Pseudo-Labeling

Once we have an annotation model, we can apply it to unlabeled datasets and use the generated pseudo-labels to further augment the annotation model. This is a technique called pseudo-labeling in semi-supervised learning. In our case, we design a pseudo-labeling pipeline with a carefully designed training process. Specifically, we first generate two augmented versions of the same gesture data, one stronger and one weaker. We apply our seed annotation model on the weakly-augmented data and generate pseudo-labels. This is an easier task as the data is less perturbed. Then, for those labels with high confidence, we assigned them to the strongly-augmented version of the data (a harder task) and further trained the annotation model. By progressing to a more challenging task, the model can further improve its performance.

Formally, let $X = \{(\mathbf{x}_b, \mathbf{y}_b : b \in (1, ..., B))\}$ be a batch of B labeled examples, where $\mathbf{x}_b$ are the training examples and $\mathbf{y}_b$ are labels. Let $U = \mathbf{u}_b : b \in (1, ..., \mu B)$ be a batch of μB unlabeled examples where μ is a hyperparameter that determines the relative sizes of X and U. Let $f_{seq2seq}$ denotes our proposed seq2seq model. Our approach computes a pseudo-label for each unlabeled sample which is then used in a CTC loss function. We perform two types of augmentations: strong and weak, denoted by $\mathrm{A}(\cdot)$ and $\alpha(\cdot)$.

We denote the CTC loss function (not in the negative log domain) as $p_{CTC}(\mathbf{y}, \mathbf{x})$ from Equation 1 between label $\mathbf{y}$ and input $\mathbf{x}$. Let $\mathbf{M}_b = f_{seq2seq}(\alpha(\mathbf{u}_b))$ be the class-conditional probability distribution produced by the model given a weakly-augmented version of a given unlabeled sample: $\alpha(\mathbf{u}_b)$. Then we use $\hat{\mathbf{y}}_b = \textit{path\_search}(\mathbf{M}_b)$ as a pseudo-label (*path_search* is our greedy search algorithm in the decoding step in Sec. IV-B.4). We enforce the CTC loss against the model's output for a strongly-augmented version of the unlabeled sample: $\mathrm{A}(\mathbf{u}_b)$:

$$L = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{I}(\textit{LP\_path\_search}(q_b) \geq \tau) p_{CTC}(\hat{\mathbf{y}}_b \mid, \mathrm{A}(\mathbf{u}_b))$$

where *LP_path_search* is the log-probabilities of the best path in greedy search, and τ is the threshold,

We employ basic time-series transformations [15] for data augmentation, which include scaling, shifting, time interpolation, and adding noise. These transformations help to diversify the training data and enhance the robustness of the model. For weak augmentation, we only apply noise; for strong augmentation, we apply all the transformations.

The threshold for determining if the pseudo-labels should be remained and be used to update the model controls the trade-off between the quality and the quantity of pseudo-labels. Recent work suggests that the quality of pseudo-labels is more important than the quantity for a good performance [48]. Thus we empirically set the threshold to 0.9.

V. Experiments and Analysis

We conducted a series of experiments to evaluate our proposed annotation model and the framework.

	SHREC'2021 [3]	**Online DHG** [5]
Static	one - four, OK, menu	NA
Dynamic	left, right, circle, v, cross	rotation (counter-)clockwise, right/left, up/down, cross, plus, v, shake
Fine Dynamic	grab, pinch, tab, deny, knob, expand	grab, pinch, tab, expand
Size	180 seqs of 3-5 gestures occurring sequentially	280 seqs of 10 gestures occurring sequentially

TABLE I: Contents of the datasets. Static gestures are characterized by keeping a fixed hand pose for a minimum amount of time, while dynamic gestures are characterized by a single trajectory with an unchanged hand pose or with finger articulation over time. Fine dynamic gestures are characterized by a single trajectory with changing hand pose.

A. Evaluation Datasets

In order to ensure a thorough evaluation of our framework, we perform our experiments on two publicly available datasets. These datasets consist of sequences of unsegmented gesture data, encompassing various types of gestures. The specific details of the two datasets are as follows:

1) **SHREC'2021 [3]**: The dataset utilized in this study consists of 18 distinct gesture classes belonging to various types. It comprises a total of 180 gesture sequences, each carefully designed to incorporate 3 to 5 gestures, accompanied by additional semi-random hand movements labeled as non-gestures. The original dictionary comprises 18 gestures, encompassing both static and dynamic gestures.
2) **Online DHG [5]**: The dataset used in this study comprises 14 distinct gesture classes from various categories. It consists of 280 sequences where each sequence contains 10 unsegmented gestures occurring sequentially. The data in this dataset is represented in the form of skeleton data, specifically capturing 22 joints for each hand skeleton. The skeleton data was collected using a Leap Motion device.

Both datasets already provided a predefined split between training and testing data. Thus, we conducted evaluations on the designated testing sets.

B. Evaluation Metrics

We introduce our metrics to evaluate the tasks of gesture classification and localization.

1) **Gesture Classification Accuracy**: Gesture annotation would generate a sequence of classification output. We use the Levenshtein distance (also known as minimum edit distance) to evaluate the gesture classification performance [18]. It is a metric defined as the minimum number of single-character (in our case, classification output of each frame) insertions, deletions, and substitutions required to transform one string into another. The accuracy for recognition performance is defined as:

$$1 - \frac{\mathrm{levenshtein}(y_{predict}, y_{true})}{\mathrm{length}(y_{true})} \tag{3}$$

	SHREC'2021 [3]		Online DHG [5]	
	Accuracy	NNLE	Accuracy	NNLE
A-ResLSTM [43] w/ CE Loss (Baseline Model)	88.3 (± 2.34)	0.42 (± 0.07)	89.8 (± 3.12)	0.44 (± 0.03)
A-ResLSTM w/ CTC loss	90.6 (± 1.45)	0.17 (± 0.03)	91.5 (± 3.35)	0.16 (± 0.04)
A-ResLSTM w/ CTC loss & many2many	91.9 (± 1.12)	0.14 (± 0.03)	92.7 (± 3.35)	0.15 (± 0.04)
A-ResLSTM w/ CTC loss & many2many & dynamic adjustment	**92.6 (± 1.58)**	**0.12 (± 0.02)**	**93.2 (± 1.76)**	**0.11 (± 0.03)**

TABLE II: Ablation study results by adding our multiple design parts step by step. NNLE is our metric for gesture localization (**N**ormalized **N**ucleus **L**abeling **E**rror). The numbers in the brackets indicate std.

where $y_{predict}$ and y_{true} are the predicted and true list of labels of the gestures, respectively.

2) **Nucleus Localization Error**: We propose a metric Normalized Nucleus Localization Error (NNLE) to measure how accurately our model can locate the gesture nucleus. When a gesture is recognized, we define the time for the start/end of the gesture as idx_{start}/idx_{end}, and the detected location of gesture nucleus as $idx_{nucleus}$. For a accurate nucleus localization, $idx_{start} \leq idx_{nucleus} \leq idx_{end}$. NNLE is defined as:

$$\frac{idx_{activation} - (idx_{start} + idx_{end})/2 + 1}{idx_{end} - idx_{start} + 1} \quad (4)$$

A smaller NNLE means our annotation model is more accurate for gesture localization (i.e., the nucleus is closer to the center of the gesture).

C. Training Details

Our implementation was based on TensorFlow 2. We utilized the Adam optimizer [17] with a starting learning rate of 0.0001. To prevent overfitting, we employed early stopping with a patience of 5. For both the Online DHG and SHREC'2021 datasets, we used a window length of 200 for our model. It is important to note that the length of an individual gesture within the datasets ranged from 20 to 50 frames, corresponding to a duration of approximately 0.4 to 1 second.

D. Ablation Study

Our novel annotation model essentially consists of multiple design blocks: (1) using the CTC loss instead of a basic CE loss, (2) using a many-to-many architecture (i.e., seq2seq) instead of a many-to-one architecture, and (3) using dynamic adjustment on labeling and nucleus localization. To evaluate the effectiveness of each design block, we performed an ablation study through a step-by-step addition process.

Table II illustrates the ablation study results. Compared to the baseline (A-ResLSTM trained with Cross-Entropy (CE) loss), our model significantly reduces the gesture localization error by 0.30 (71.4%) and 0.33 (75.0%). Regarding gesture classification accuracy, our model design achieves improvements of 4.3% on the SHREC'2021 dataset and 3.4% on the Online DHG dataset compared to the baseline.

Each of the three blocks contributes approximately equally to the overall gesture classification accuracy improvement. For gesture nucleus localization, the main contribution stems from the application of the Connectionist Temporal Classification (CTC) loss. This highlights the unique character of the CTC-trained model in producing outputs with a spiky nature, eliminating the need for complex post-processing techniques (see Figure 3). In contrast, CE-trained networks require post-processing to determine the final label, often relying on heuristic methods.

E. Annotation Framework Evaluation

To evaluate the effectiveness of our framework, we incorporate a similar pseudo-labeling process into the baseline model described in Section V-D. This modified version serves as a baseline framework for comparison.

1) Baseline: It is important to note that the pseudo labeling semi-supervised learning pipeline we developed specifically for our proposed annotation model cannot be directly applied to the baseline model, as the baseline model is trained using Cross-Entropy (CE) loss. Therefore, we made modifications to adapt the pseudo-labeling process to the baseline model, enabling a fair comparison between the two frameworks.

Formally, we denote the CE loss function between two probability distributions p and q as $H(p, q)$. Let $q_b = p(\mathbf{y} \mid \alpha(\mathbf{u}_b))$ be the class-conditional probability distribution produced by the model given a weakly-augmented version of a given unlabeled sample: $\mathbf{u}$. Then we use $\hat{q}_b = \arg\max(q_b)$ through the path decoding algorithm as a pseudo-label, and we enforce the CE loss against the model's output for a strongly-augmented version of $\mathbf{u}_b$:

$$L = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{I}(\arg\max(q_b) \geq \tau) H(\hat{q}_b \mid, p(\mathbf{y} \mid \mathrm{A}(\mathbf{u}_b)))$$

where τ is the threshold. The rest of the semi-supervised learning pipeline stays the same as in Section IV-C.

2) Procedure: The ultimate objective of our framework is to generate high-quality pseudo-labels that can effectively fine-tune and enhance downstream gesture recognition models. To evaluate the efficacy of our pipeline, we measure the performance improvement achieved by fine-tuning the initial pre-trained gesture recognition model using our generated pseudo-labels.

In the experiments, we create a subset of the training dataset that initially contains complete labels. From this subset, we remove the labels and leverage our framework to generate pseudo-labels for this particular portion of the data. The size of the remaining subset of labeled data becomes the

	SHREC'2021 [3]			Online DHG [5]		
Labeled Sequences	~ 40	~ 80	~ 120	~ 70	~ 140	~ 210
Baseline [43]	**30.0 (Δ=22%)**	69.0 (Δ=11%)	83.3 (Δ=6%)	**34.9 (Δ=25%)**	68.3 (Δ=10%)	81.6 (Δ=5%)
Our Annotation Framework	26.0 (Δ=18%)	**73.0 (Δ=15%)**	**88.3 (Δ=11%)**	28.9 (Δ=19%)	**75.3 (Δ=17%)**	**89.6 (Δ=13%)**

TABLE III: Evaluation of the effectiveness of the pseudo-labeled dataset produced from our proposed annotation framework. The numbers indicate the gesture classification accuracy of the downstream gesture recognition model fine-tuned from the pseudo-labeled dataset. The Δ indicates the accuracy improvement between the pre-trained accuracy and the fine-tuned accuracy. The second row indicates the size of the labeled subset. These labeled sequences are used for pre-training the downstream gesture recognition model and the annotation model. Please refer to Figure 1 for a detailed illustration of the workflow of the annotation framework. Note that SHREC'2021 dataset has a total of 180 sequences, and Online DHG dataset has a total of 280 sequences.

adjustable parameter in our experiments. Now, we have two subsets, one labeled subset and one unlabeled subset. This labeled subset is utilized for pre-training both the annotation model and the downstream gesture recognition model, and the unlabeled subset is utilized in the pseudo labeling-based semi-supervised learning pipeline. Such a process is demonstrated in Figure 1. We use one of the state-of-the-art gesture recognition models, ST-GCN, from [3] as the downstream gesture recognition model. The performance of the framework directly impacts the quality of the pseudo-labels produced from the annotation framework. The quality of the pseudo-labels then impacts the improved accuracy and final accuracy of the final downstream recognition model which is fine-tuned from the pseudo-labels. This is explained in Figure 1. We use gesture classification accuracy in Equation 3 to measure the accuracy of the downstream gesture recognition model.

3) Results: The performance and improvements of the fine-tuned downstream recognition model are summarized in Table III. It is important to note that these results pertain to the downstream model and are not directly comparable to those in Table II.

Through the semi-supervised learning pipeline, both our new annotation framework and the baseline framework demonstrate significant improvements in the performance of the downstream model, as indicated by the Δ values, from 5% to 25%. The magnitude of improvement is directly influenced by the size of the unlabeled dataset, with larger datasets yielding greater improvements through this pipeline.

Furthermore, our evaluation results highlight that the pseudo-labeled data generated by our framework generally outperforms the pseudo-labeled data generated by the baseline framework. The quality of the pseudo labels is reflected in both the improved accuracy and the final accuracy of the downstream model. It is noteworthy that the baseline framework initially outperforms our new framework when using a small set of labeled data. This can be attributed to the higher number of trainable parameters in the output layer of our annotation model, which employs a many2many design and faces challenges in achieving convergence with the CTC loss. However, as the size of the labeled set increases, our new framework surpasses the baseline.

The superior performance of our framework can be attributed to the annotation model's ability to train on pseudo-labeled data that may not be accurately segmented initially in the semi-supervised learning loop (pipeline). This is in contrast to the baseline framework, whose model heavily relies on accurately segmented data when trained with CE loss. These results highlight the potential of our annotation model and the semi-supervised pipeline to significantly enhance downstream gesture recognition models, showcasing the advantages of our approach.

VI. Limitation and Future Work

It is important to acknowledge several limitations in our work. Firstly, although our proposed model exhibits superior performance in gesture data annotation for class labeling and nucleus localization, we must acknowledge that the training process of our new model is more time-consuming compared to the baseline. While training efficiency is not the primary focus since annotation can occur offline, there is room for future improvement in this aspect. Furthermore, our current framework still relies on the availability of labeled data to initiate the process. We have not explored a fully unsupervised version in this study, which presents an intriguing avenue for future research.

VII. Conclusion

This paper presents a framework that enables accurate and automated annotation of gesture data, encompassing gesture classification and localization. The framework consists of two key components: (1) a novel annotation model that optimizes both gesture classification and localization concurrently, and (2) a semi-supervised learning pipeline incorporating pseudo-labeling. The ablation study reveals that this novel annotation model design surpasses the baseline model, achieving a 4.3% higher class labeling accuracy and a 71.4% improvement in nucleus localization accuracy on the SHREC'2021 dataset (3.4% and 75.0% respectively on the Online DHG dataset). Moreover, we also demonstrate that the pseudo-labeled data generated by our framework significantly enhances the performance of a pre-trained downstream gesture recognition model through fine-tuning, resulting in improvements ranging from 11% to 18% on the SHREC'2021 and Online DHG datasets.

REFERENCES

[1] P. Bachman, O. Alsharif, and D. Precup. Learning with pseudo-ensembles. *Advances in neural information processing systems*, 27, 2014.

[2] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2020.

[3] A. Caputo, A. Giachetti, S. B. Soso, D. Pintani, A. D'Eusanio, S. Pini, G. Borghi, A. Simoni, R. Vezzani, R. Cucchiara, A. Ranieri, F. Giannini, K. Lupinetti, M. Monti, M. Maghoumi, J. J. Laviola, M.-Q. Le, H.-D. Nguyen, and M. Tran. Shrec 2021: Skeleton-based hand gesture recognition in the wild. *Comput. Graph.*, 2021.

[4] Y. Chen, L. Zhao, X. Peng, J. Yuan, and D. N. Metaxas. Construct dynamic graphs for hand gesture recognition via spatial-temporal attention. In *British Machine Vision Conference*, 2019.

[5] Q. De Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. Le Saux, and D. Filliat. SHREC'17 Track: 3D Hand Gesture Recognition Using a Depth and Skeletal Dataset. In *3DOR - 10th Eurographics Workshop on 3D Object Retrieval*, 2017.

[6] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[7] O. Duchenne, I. Laptev, J. Sivic, F. Bach, and J. Ponce. Automatic annotation of human actions in video. In *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009.

[8] B. Feng, F. He, X. Wang, Y. Wu, H. Wang, S. Yi, and W. Liu. Depth-projection-map-based bag of contour fragments for robust hand gesture recognition. *IEEE Transactions on Human-Machine Systems*, 47:511–523, 2017.

[9] N. E. Gillian and J. A. Paradiso. The gesture recognition toolkit. In *Journal of machine learning research*, 2014.

[10] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, 2006.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[12] S. Hochreiter and J. Schmidhuber. Lstm can solve hard long time lag problems. *Advances in neural information processing systems*, 1997.

[13] F. Hu, P. He, S. Xu, Y. Li, and C. Zhang. FingerTrak: Continuous 3D Hand Pose Tracking by Deep Learning Hand Silhouettes Captured by Miniature Thermal Cameras on Wrist. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2020.

[14] N. Ienaga, A. Cravotta, K. Terayama, B. Scotney, H. Saito, and M. G. Busa. Semi-automation of gesture annotation by machine learning and human collaboration. *Language Resources and Evaluation*, 2022.

[15] B. K. Iwana and S. Uchida. An empirical survey of data augmentation for time series classification with neural networks. *PLoS ONE*, 2020.

[16] A. Jaramillo-Yánez, M. E. Benalcázar, and E. Mena-Maldonado. Real-time hand gesture recognition using surface electromyography and machine learning: a systematic literature review. *Sensors*, 2020.

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, 2014.

[18] O. Köpüklü, A. Gunduz, N. Kose, and G. Rigoll. Real-time hand gesture detection and classification using convolutional neural networks. In *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*. IEEE, 2019.

[19] S. Kratz and M. Back. Towards accurate automatic segmentation of imu-tracked motion gestures. CHI EA '15. Association for Computing Machinery, 2015.

[20] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

[21] G. Laput, R. Xiao, and C. Harrison. ViBand: High-Fidelity Bio-Acoustic Sensing Using Commodity Smartwatch Accelerometers. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 2016.

[22] D.-H. Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, 2013.

[23] J. Liu, Y. Liu, Y. Wang, V. Prinet, S. Xiang, and C. Pan. Decoupled representation learning for skeleton-based gesture recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[24] J. Liu, A. Shahroudy, D. Xu, and G. Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European conference on computer vision*. Springer, 2016.

[25] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[26] J. McIntosh, A. Marzo, and M. Fraser. SensIR: Detecting Hand Gestures with a Wearable Bracelet using Infrared Transmission and Reflection. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, Québec City QC Canada, Oct. 2017. ACM.

[27] Y. Min, Y. Zhang, X. Chai, and X. Chen. An efficient pointlstm for point clouds based gesture recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[28] Z. Mo and U. Neumann. Real-time hand pose recognition using low-resolution depth images. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2:1499–1505, 2006.

[29] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[30] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[31] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic. Deep learning applications and challenges in big data analytics. *Journal of big data*, 2015.

[32] X. S. Nguyen, L. Brun, O. Lézoray, and S. Bougleux. A neural network based on SPD manifold learning for skeleton-based hand gesture recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12036–12045, 2019.

[33] G. Park, T.-K. Kim, and W. Woo. 3D Hand Pose Estimation with a Single Infrared Camera via Domain Transfer Learning. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 588–599, Nov. 2020. ISSN: 1554-7868.

[34] D. Ramanan and D. Forsyth. Automatic annotation of everyday movements. *Advances in neural information processing systems*, 2003.

[35] S. Reifinger, F. Wallhoff, M. Ablassmeier, T. Poitschke, and G. Rigoll. Static and dynamic hand-gesture recognition for augmented reality applications. In *International Conference on Human-Computer Interaction*. Springer, 2007.

[36] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 2021.

[37] Z. Ren, J. Yuan, and Z. Zhang. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. *Proceedings of the 19th ACM international conference on Multimedia*, 2011.

[38] C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*, 1:29–36, 2005.

[39] K. M. Sagayam and D. J. Hemanth. Hand posture and gesture recognition techniques for virtual reality applications: a survey. *Virtual Reality*, 21(2):91–107, 2017.

[40] M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 2016.

[41] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

[42] J. Shen, J. Dudley, and P. O. Kristensson. The imaginative generative adversarial network: Automatic data augmentation for dynamic skeleton-based hand gesture and human action recognition. In *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*, page 1–8. IEEE Press, 2021.

[43] J. Shen, J. J. Dudley, M. George, and P. O. Kristensson. Gesture Spotter: A Rapid Prototyping Tool for Key Gesture Spotting in Virtual and Augmented Reality Applications. In *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[44] J. Shen, J. J. Dudley, and P. O. Kristensson. Simulating realistic human motion trajectories of mid-air gesture typing. *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 393–402, 2021.

[45] J. Shen, J. J. Dudley, and P. O. Kristensson. Fast and robust mid-air gesture typing for ar headsets using 3d trajectory decoding. *IEEE*

Transactions on Visualization and Computer Graphics, 2023.

[46] J. Shen, M. D. Lange, X. O. Xu, E. Zhou, R. Tan, N. Suda, M. Lazarewicz, P. O. Kristensson, A. Karlson, and E. Strasnick. Towards open-world gesture recognition. *ArXiv*, abs/2401.11144, 2024.

[47] X. Shi, J. Pan, Z. Hu, J. Lin, S. Guo, M. Liao, Y. Pan, and L. Liu. Accurate and Fast Classification of Foot Gestures for Virtual Locomotion. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Oct. 2019.

[48] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 2020.

[49] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 2017.

[50] M. Vandersteegen, W. Reusen, K. Van Beeck, and T. Goedemé. Low-latency hand gesture recognition with a low-resolution thermal imager. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition workshops*, 2020.

[51] H. Wen, J. Ramos Rojas, and A. K. Dey. Serendipity: Finger Gesture Recognition using an Off-the-Shelf Smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016.

[52] E. Wu, Y. Yuan, H.-S. Yeo, A. Quigley, H. Koike, and K. M. Kitani. Back-Hand-Pose: 3D Hand Pose Estimation for a Wrist-worn Camera via Dorsum Deformation Network. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, Virtual Event USA, 2020.

[53] Q. Xie, Z. Dai, E. Hovy, T. Luong, and Q. Le. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 2020.

[54] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[55] X. Xu, A. Dancu, P. Maes, and S. Nanayakkara. Hand range interface: information always at hand with a body-centric mid-air input surface. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*, Barcelona Spain, 2018. ACM.

[56] X. Xu, J. Gong, C. Brum, L. Liang, B. Suh, S. K. Gupta, Y. Agarwal, L. Lindsey, R. Kang, B. Shahsavari, T. Nguyen, H. Nieto, S. E. Hudson, C. Maalouf, S. A. Mousavi, and G. Laput. Enabling hand gesture customization on wrist-worn devices. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, 2022.

[57] X. Xu, J. Li, T. Yuan, L. He, X. Liu, Y. Yan, Y. Wang, Y. Shi, J. Mankoff, and A. K. Dey. HulaMove: Using Commodity IMU for Waist Interaction. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–16, Yokohama Japan, May 2021. ACM.

[58] X. Xu, H. Shi, X. Yi, W. Liu, Y. Yan, Y. Shi, A. Mariakakis, J. Mankoff, and A. K. Dey. EarBuddy: Enabling On-Face Interaction via Wireless Earbuds. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, Honolulu HI USA, 2020.

[59] H.-S. Yeo, E. Wu, J. Lee, A. Quigley, and H. Koike. Opisthenar: Hand Poses and Finger Tapping Recognition by Observing Back of Hand Using Embedded Wrist Camera. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, 2019.

[60] D. Zhang, M. M. Islam, and G. Lu. A review on automatic image annotation techniques. *Pattern Recognition*, 2012.

[61] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.